A Fortran Code for Null Geodesic Solutions in the Lemaître-Tolman-Bondi Spacetime

Marcelo B. Ribeiro [1,2]

Vatican Observatory Group, Steward Observatory, University of Arizona, Tucson, AZ 85721, USA; e-mail: mribeiro@as.arizona.edu

Abstract

This paper describes the Fortran 77 code `SIMU`, version 1.1, designed for numerical simulations of observational relations along the past null geodesic in the Lemaître-Tolman-Bondi (LTB) spacetime. `SIMU` aims at finding scale invariant solutions of the average density, but due to its full modularity it can be easily adapted to any application which requires LTB's null geodesic solutions. In version 1.1 the numerical output can be read by the `GNUPLOT` plotting package to produce a fully graphical output, although other plotting routines can be easily adapted. Details of the code's subroutines are discussed, and an example of its output is shown.

Key words: cosmology, numerical simulation; solution of equations, numerical relativity
PACS: 98.80.-k, 02.60.Cb, 04.25.Dm

1 Introduction

It is a basic result of relativity theory that light rays follow null geodesics in curved 4-dimensional spacetimes, and this means that if one is pursuing cosmological applications of solutions of Einstein's field equations with the aim of comparing the model's theoretical predictions with the actual data produced by astronomical observations, one requires null geodesic solutions for the chosen metric. In fact, for cosmological applications one only needs the *past* null geodesics, as what we observe today are events which occurred in the past. Nevertheless, a very basic problem arising in cosmology, and which hinders comparison between theory and observations in other models than

---

[1] On leave from Physics Institute, University of Brazil – UFRJ, Rio de Janeiro.
[2] Supported by Brazil's CAPES Foundation.

the standard Friedmann-Lemaître-Robertson-Walker (FLRW), stems from the fact that as soon as we depart from the simple FLRW models largely used by observational cosmologists, the task of finding null geodesic solutions quickly becomes an intractable analytical problem. In fact, this situation is not fully appreciated by many of those dealing with cosmological modelling of astronomical data, inasmuch as the simple FLRW models largely applied in observational cosmology is the exception, since it has a very simple analytical solution for the null geodesic equation.

The Lemaître-Tolman-Bondi (LTB) spacetime is the most general spherically symmetric dust solution of Einstein's field equations, having the FLRW models as special sub-cases [1–3]. By being a spatially inhomogeneous model, that is, where $\rho = \rho(r, t)$, it allows us to study scenarios which do not have FLRW's spatial homogeneity assumed *a priori*. Due to this, it is the most widely used cosmological model after the standard FLRW, having been applied in a wide range of cosmological problems, from microwave background radiation studies to hierarchical (fractal) modelling, just to quote a few (see [3], and references therein, for a large number of applications of LTB models in cosmology). However, as is the case with almost all non-standard cosmologies, while it is possible to solve analytically its Einstein's field equations, LTB's null geodesic equation remains an intractable analytical problem, unless we rewrite its geometry in terms of the so-called "observational coordinates". That, however, has the handicap of adding a great deal of mathematical complexity to the problem, since such an approach requires complex mathematical calculations in curved spacetimes [4], something which may not be required in all LTB applications. Therefore, due to such a wide range of application, a code which produces numerical solutions of LTB null geodesics is desirable.

In this paper I will describe such a code. It deals with LTB geometry in its full generality, being applicable to each of its special sub-cases, parabolic, hyperbolic and elliptic [5], either separately or together in a single problem encompassing all sub-cases, if desired. It was originally designed to find scale invariant solutions of the average density by means of numerical simulations [6,7], but as we shall see below, without changing its core null geodesic calculations the code can be easily adapted to do much more than this, due to its modularity. The original results obtained by this code were recently analytically confirmed, and extended, by other authors [8] by means of the observational coordinates approach mentioned above. The confirmation of the results of [7] by [8] adds then further reliability to the numerical results obtained with the code that will be described below. Besides, those two approaches, numerical and analytical, when applied to a difficult problem such as null geodesic solutions in LTB spacetime, will in fact complement each other, rather than exclude one another. In §2 I will describe version 1.1 of the code, and in §3 the input and output will be discussed. The results are summarized in §4, where I also indicate where the code can be obtained.

2 Description of SIMU 1.1

SIMU solves simultaneously two ordinary differential equations (ODE's) by means of the fourth order Runge-Kutta method with adaptive stepsize control. One differential equation is required for the null geodesic, while the other is needed for obtaining the redshift in this cosmology [6]. The subroutines that carry out the numerical integration are from [9], and some minor changes necessary for taking the results back to the main program were made, but without changing the actual numerical implementation of the Runge-Kutta method. A full description of the LTB model and notation as used in SIMU is given in [6]. The numerical simulations take advantage of the fact that LTB spacetime geometry has three unknown functions,[3] $F(r)$, $f(r)$, $\beta(r)$, respectively representing the amount of gravitational mass within the radius coordinate r, the overall curvature and dynamics of the model, and the time elapsed since the big bang for each observer located at particular values of r [2,3,5]. The simulations then take advantage of this freedom, as one starts by choosing these functions, run the program and analyse the results, concluding then whether or not the simulation was successful, and if not choose another set of three functions (see [7] for a very detailed explanation of this procedure).

The code is implemented in double precision, and it has a built-in methodology for checking the possible catastrophic accumulation of round-off errors. That is done by monitoring the energy equation, as derived from Einstein's field equations, and its derivative, since, by theory, both must remain unchanged throughout the integration [7]. In addition, to check for accuracy and stability, after the integration the program runs in reverse, that is, it takes the final result and uses it as initial conditions to get another result that can be compared with the original initial condition. The results described in [7] showed that double precision is enough for maintaining the desired accuracy required by the problem under study. SIMU also calculates the errors associated with each observational quantity evaluated, as well as the propagated errors. This is done by taking the accuracy given by the adaptive method and using this value as the input in the standard propagation error equations. All those details are fully explained in the initial documentation of the code.

As mentioned above, the code uses some subroutines from [9], namely the fourth order Runge-Kutta integration with adaptive stepsize control, the root finding algorithm for the transcendental equations appearing in LTB's elliptical and hyperbolic sub-cases [6], the extended trapezoidal rule quadrature algorithm, as well as Simpson's rule to the desired accuracy. However, apart from those, the remaining subroutines are new, as well as the actual way in which the whole set of subroutines were linked to each other and organized.

---

[3] Actually two, as a third can be eliminated by a coordinate transformation.

In addition, the code is extensively commented and widely documented, also having a structure chart showing the dependence among the subroutines and functions. This detailed documentation is added in order to help readers who might be interested in implementing `SIMU` in their own computer environments, or changing it to suit their specific applications.

It must be mentioned that there are more "state of the art" ways for finding numerical solutions of ODE's than using the adaptive 4th order Runge-Kutta method. The ODEPACK subroutine package is an example [10–12]. Nevertheless, although the chosen ODE integrator for `SIMU` may have already been superseded by better methods, it *does* solve the proposed problem, and at the desired accuracy. Inasmuch as `SIMU` is a tested code, with its results having already been analytically confirmed by other authors [8], there is no need to change to another ODE integrator at `SIMU`'s current version. The reader must, however, be aware that such a change might be required for other applications of `SIMU` to LTB spacetime.

The code uses only standard Fortran 77 commands to avoid possible compilation problems with machine dependent commands that are not available universally. If a potential user is only interested in the core of the program, that is, the part which integrates the null geodesic and calculates the observational relations, he or she can simply remove its "tail", where the observational relations are manipulated according to the aims of the theory studied in [6,7], and replace it with something else. This "tail" consists of the two last branches of the structure chart headed by the subroutines `FIT` and `OUTPUT`, and are easily spotted in the `MAIN`. Finally, the functions $f(r)$, $F(r)$, $\beta(r)$ and their derivatives appear in six subroutines at the end of the list so that they can be compiled separately.

The previous version 1.0 (formerly `SIMU 5d`; see [13]) had the `PGPLOT` plotting routines merged into the second half of the `OUTPUT` subroutine, while version 1.1 removed them and piped the numerical results into eleven independent files, `v1` to `v11`, that are then *externally* read by the `GNUPLOT` plotting package in order to produce a graphical output of the results. This is the only difference between versions 1.0 and 1.1, meaning that in version 1.1 the graphical output is completely independent from the code itself. Making this change was justified on two grounds: *(i)* the `GNUPLOT` package has greater plotting capabilities, even allowing LaTeX output formats that can be included directly in the `figure` environment of any LaTeX macro. So, producing plots with the results is done by feeding the numerical tables into `GNUPLOT` itself by means of a script file written for, say, a linux environment; *(ii)* this feature of having the results outputted into numerical files brings flexibility to the code, as its results can be independently used by another program, if so desired or demanded by another application of the LTB spacetime.

3 Input and Output

The actual implementation for the original problem advanced in [6] was described at length in [7], where one can find detailed plots with the results and analysis of the various simulations,[4] as well as detailed explanations of initial conditions and functions used in each simulation. So, in here I shall limit the discussion to the code itself, without dealing with any specific application.

As mentioned above, we require a script file for running a simulation and immediately producing a graphical output. An example of such a script in linux is given as follows:

```
g77 simu1.1.for
a.out < in.simu
gnuplot < in.gnu
latex plots.tex
xdvi plots.dvi
```

Here the file `in.simu` contains the maximum value for the radius coordinate r for the integration, while `in.gnu` is a short GNUPLOT script file for inputting the `v1` to `v11` files and producing LaTeX files (see appendix A). Finally `plots.tex` is a file for running the `.tex` LaTeX files produced by GNUPLOT in order for the results be available for analysis directly on the screen.

The details of each simulations, as well as accuracy monitoring are given in an outputted file called `s1`. Appendix B shows the results of a simulation using the value 1.5 for `in.simu`, and the three functions given as follows: $f(r) = \cosh(r)$, $F(r) = \sinh^3(r)$, $\beta(r) = 0.7$.

4 Conclusion

In this paper I have described the Fortran 77 code `SIMU` version 1.1 for calculating solutions of the null geodesics equations in the Lemaître-Tolman-Bondi spacetime geometry. I have discussed the details of the code, input and output, as well as the ways in which it can be possibly modified for other cosmological applications of this geometry. The code is available for download at [14].

---

[4] These results were summarized in [2].

Acknowledgements

References

[1] M. B. Ribeiro, *Astrophys. J.* **395** (1992) 29

[2] M. B. Ribeiro, *in:* D. Hobill, A. Burd and A. Coley, eds., *Deterministic Chaos in General Relativity* (New York, Plenum, 1994) 269-296

[3] A. Krasiński, *Inhomogeneous Cosmological Models* (Cambridge University Press, 1997)

[4] R. Marteens, N. P. Humphreys, D. R. Matravers and W. R. Stoeger, *Class. Quantum Grav.* **13** (1996) 253; corrigenda **13** (1996) 1689; M. E. Araújo and W. R. Stoeger, *Phys. Rev. D* **60** (1999) 104020

[5] W. B. Bonnor, *Monthly Not. Royal Astron. Soc.* **167** (1974) 55

[6] M. B. Ribeiro, *Astrophys. J.* **388** (1992) 1

[7] M. B. Ribeiro, *Astrophys. J.* **415** (1993) 469

[8] N. P. Humphreys, D. R. Matravers and R. Marteens, *Class. Quantum Grav.* **15** (1998) 3041, gr-qc/9804025

[9] W. H. Press, B. P. Flannery, S. A. Teukolsky and W. T. Vetterling, *Numerical Recipes*, (Cambridge University Press, 1986)

[10] A. C. Hindmarsh, *'ODEPACK, a Systematized Collection of ODE Solvers'*, *in:* R. S. Stepleman et al., eds., *Scientific Computing* (North-Holland, 1983) 55-64

[11] L. R. Petzold, *SIAM J. Sci. Sta. Comput.* **4** (1983) 136

[12] L. R. Petzold and A. C. Hindmarsh, ODEPACK subroutine package available from netlib at `http://www.netlib.org/odepack/`

[13] M. B. Ribeiro, *On Modelling a Relativistic Hierarchical (Fractal) Cosmology by Tolman's Spacetime*, PhD thesis, (Queen Mary and Westfield College, University of London, 1992); `http://www.if.ufrj.br/~mbr/publi.html`

[14] `SIMU 1.1` can be downloaded from `http://www.if.ufrj.br/~mbr/codes` or `http://arXiv.org/abs/gr-qc/0205`

A in.gnu Script

```
set terminal latex
set output 'v1.tex'
set size 1,1
set xlabel "$ \log d_l (Gpc)$"
set ylabel "$ \log \rho$"
set nokey
plot 'v1' with points 1 8, 'v11' with lines 1
set output 'v2.tex'
set xlabel "$d_l (Gpc)$"
set ylabel "$ \rho$"
plot 'v2' with points 1 8
set output 'v3.tex'
set xlabel "$z$"
set ylabel "$d_l$ \\ $(Gpc)$"
plot 'v3' with points 1 8
set output 'v4.tex'
set xlabel "$z$"
set ylabel "$N_c$"
plot 'v4' with points 1 8
set output 'v5.tex'
set xlabel "$r$"
set ylabel "$Field$ \\ $EE$"
set yrange [-10**-5:10**-5]
plot 'v5' with points 1 8
set output 'v6.tex'
set xlabel "$r$"
set ylabel "$Deriv.$ \\ $Field$ \\ $EEDSH$"
set yrange [-10**-5:10**-5]
plot 'v6' with points 1 8
set output 'v7.tex'
set title "$Center \rightarrow Border$"
set xlabel "$r$"
set ylabel "$t$"
set autoscale y
plot 'v7' with points 1 8
set output 'v8.tex'
set title "$Border \rightarrow Center$"
set xlabel "$r$"
set ylabel "$t$"
plot 'v8' with points 1 8
set output 'v9.tex'
set title "$Center \rightarrow Border$"
```

```
set xlabel "$r$"
set ylabel "$I$"
plot 'v9' with points 1 8
set output 'v10.tex'
set title "$Border \rightarrow Center$"
set xlabel "$r$"
set ylabel "$I$"
plot 'v10' with points 1 8
quit
```

B   Plots of a Single Simulation

$Center \rightarrow Border$

$Border \rightarrow Center$

$Center \rightarrow Border$

$Border \rightarrow Center$